INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA

I MARATONA DE PROGRAMAÇÃO INTERIF – 2018

Caderno de Problemas

Informações Gerais

Este caderno contém 8 problemas, as páginas estão numeradas de 1 a 12, não contando esta página de rosto. Verifique se o caderno está completo.

A) Aquecimento

1) O primeiro exercício (problema A) será utilizado como aquecimento.
2) As soluções se encontram neste caderno de problemas nas três linguagens: C, C++ e Java.
3) Os times deverão digitar e submeter este programa na linguagem que preferir.
4) A resolução deste exercício será utilizada para a classificação final.

B) Sobre a entrada

1) A entrada de seu programa deve ser lida da entrada padrão.
2) A entrada é composta de um ou mais casos de teste, depende do problema.
3) Quando uma linha da entrada contém vários valores, estes são separados por um único espaço em branco; a entrada não contém nenhum outro espaço em branco.
4) Cada linha, incluindo a última, contém o caractere final-de-linha.
5) O final da entrada pode coincidir com o final do arquivo ou com uma entrada determinada

C) Sobre a saída

1) A saída de seu programa deve ser escrita na saída padrão.
2) Espaços em branco só devem ser colocados quando solicitado.
3) Cada linha, incluindo a última, deve conter o caractere final-de-linha.

D) Regras

1) Só é permitida a comunicação entre os membros de um mesmo grupo.
2) Não é permitido o acesso à internet.
3) Não é permitido o uso de qualquer aparelho de comunicação.
4) Não é permitido o uso de meios de armazenamento externos (*pendrives,* cartões de memória, hd externo, etc).
5) É permitida a consulta de qualquer material impresso.

Caso uma destas regras seja descumprida, a equipe será desclassificada da competição.

Problema A (Aquecimento – Resolvido)
Funções Matemáticas

Por URI UTFPR
Arquivo: funcoes.[c|cpp|java]
***Timelimit:** 1*

Na última aula de matemática, Rafael, Beto e Carlos aprenderam algumas novas funções matemáticas.

Cada um deles se identificou com uma função em especial, e resolveram competir para ver quem tinha a função de maior resultado.

A função que Rafael escolheu é $r(x, y) = (3x)^2 + y^2$.

Já Beto escolheu a função $b(x, y) = 2(x^2) + (5y)^2$.

Carlos, por sua vez, escolheu a função $c(x, y) = -100x + y^3$.

Dados os valores x e y, diga quem escolheu a função com o maior resultado.

Entrada

Cada caso de teste consiste em dois inteiros x e y ($1 \leq x, y \leq 100$), indicando as variáveis a serem inseridas na função.

Saída

Para cada caso de teste imprima uma linha, contendo uma frase, indicando quem ganhou a competição.

Por exemplo, se Rafael ganhar a competição, imprima "Rafael ganhou". Assuma que nunca haverá empates.

O resultado de seu programa deve ser escrito na saída padrão.

Exemplos de Entradas	Exemplos de Saídas
5 3	Beto ganhou
2 30	Carlos ganhou
2 100	Carlos ganhou
30 20	Beto ganhou
15 5	Rafael Ganhou

Resoluções Problema A

Linguagem C

```
#include <stdio.h>
#include <math.h>
void main()
{
     int n, x, y, r, b, c;
     scanf("%d %d",&x,&y);
     r = pow((3*x),2) + pow(y,2);
     b = 2*pow(x,2) + pow((5*y),2);
     c = -100 * x + pow(y,3);
     if(b > r && b > c)
     {
          printf("Beto ganhou\n");
     }else if (r > b && r > c)
     {
          printf("Rafael ganhou\n");
     }else{
          printf("Carlos ganhou\n");
     }
}
```

Linguagem C++

```
#include <iostream>
#include <math.h>
using namespace std;
int main(void)
{
     int n, x, y, r, b, c;
     cin>>x>>y;
     r = pow((3*x),2) + pow(y,2);
     b = 2*pow(x,2) + pow((5*y),2);
     c = -100 * x + pow(y,3);
     if(b > r && b > c)
     {
          cout<<"Beto ganhou"<<endl;
     }else if (r > b && r > c)
     {
          cout<<"Rafael ganhou"<<endl;
     }else{
          cout<<"Carlos ganhou"<<endl;
     }
}
```

Linguagem Java

```
import java.io.IOException;
import java.util.Scanner;

public class funcoes {

     public static void main(String[] args) throws IOException {

          Scanner in = new Scanner(System.in);
          int x, y;
          double r, b, c;
          String linha;
          try
          {
               x = in.nextInt();
               y = in.nextInt();
               r = Math.pow((3*x),2) + Math.pow(y,2);
               b = 2*Math.pow(x,2) + Math.pow((5*y),2);
               c = -100 * x + Math.pow(y,3);
               if(b > r && b > c)
               {
                    System.out.printf("Beto ganhou\n");
               }else if (r > b && r > c)
               {
                    System.out.printf("Rafael ganhou\n");
               }else{
                    System.out.printf("Carlos ganhou\n");
                                   }
          }catch(Exception ex){
               System.exit(0);
          }
     }
}
```

Problema B
Jogo de Truco

Por Cássio Agnaldo Onodera
Arquivo: truco.[c|cpp|java]
***Timelimit:** 1*

Sérgio chegou recentemente à Birigui para fazer o curso de Engenharia da Computação no Instituto Federal de Educação, Ciência e Tecnologia de São Paulo e foi morar com seus amigos Bruno e Carlos em uma república.

Nos momentos de folga, **S**érgio, **B**runo e **C**arlos, ficam jogando truco para passar o tempo. Bruno e Carlos já conheciam este jogo de cartas, mas Sérgio ainda está aprendendo. Ele ainda se confunde com a sequência das cartas.

O truco é um jogo de cartas que se utiliza um baralho, retirando-se as cartas 8, 9 e 10 e o curinga, ou seja, utiliza-se 10 cartas para cada naipe, totalizando 40 cartas.

Como Sérgio ainda não decorou a sequência das cartas, eles optaram por jogar sem as manilhas, que são as cartas mais fortes do jogo. Desta forma a sequência das cartas da mais fraca para a mais forte ficou assim: 4, 5, 6, 7, 11 (Q-dama), 12 (J-Valete), 13 (K-rei), 1 (A-Ás), 2 e 3. Ou seja, a carta 4 é a mais fraca e a carta 3 é a mais forte.

Faça um programa para auxiliar Sérgio em um jogo de truco.

O programa deverá receber o valor N de uma carta e o programa deverá informar quais cartas são mais forte que N.

Entrada

A entrada é constituída por uma única linha que contém um número inteiro N (1, 2, 3, 4, 5, 6, 7, 11, 12 ou 13) que representa o valor da carta do adversário de Sérgio.

Saída

O programa deverá produzir uma única linha contendo os valores de todas as cartas maiores que N. Se o valor N for igual à 3, seu programa deverá retornar o número 0. Os números deverão estar separados por um espaço em branco.

Exemplos de Entradas	Exemplos de Saídas
1	2 3
11	12 13 1 2 3
13	1 2 3
3	0

Problema C
Taxi Aéreo

Por Murilo Varges da Silva
Arquivo: taxiaereo.[c|cpp|java]
***Timelimit:** 1*

Uma empresa de taxi aéreo aluga seu avião para ***P*** ou mais passageiros. Se o número de passageiros for exatamente ***P***, cada um pagará um valor ***V***. Haverá um desconto de ***D*** reais para cada passageiro que exceder ***P*** passageiros. Como a capacidade de cada aeronave é de ***C*** passageiros, qual deverá ser o número de passageiros em cada avião, a fim de que a empresa obtenha a maior receita possível, ou seja, a receita máxima? Qual o valor da receita máxima?

Dica:
Supondo que temos (**P)assageiros** = 100, (**V)alor** = R$ 500,00, (**D)esconto** = R$ 10,00 e (**C)apacidade** = 150.
Seja **R= Receita**. Logo, **R** = Número de passageiros vezes pagamento por passageiro. Se o número de passageiros passar de 100 para 101, então:

Pagamento por passageiro = 500–10(101–100) = 500–10(1).

Se o número de passageiros passar de 100 para 102, então:

Pagamento por passageiro = 500–10 (102–100) = 500–10(2). E assim por diante.

Se o número de passageiros for ***x***, então:

Pagamento por passageiro = 500–10(x–100)=500–10x+1000=1500–10x. Como ***x*** corresponde ao número de passageiros, e a receita é igual ao número de passageiros vezes pagamento por passageiro, logo:

$$\mathbf{R} = x(1500 - 10x) \text{ ou } \mathbf{R}(\boldsymbol{x}) = -10x^2 + 1500x$$

Vamos achar o valor de ***x*** que dá o máximo à $\mathbf{R}(\boldsymbol{x}) = -10x^2 + 1500x$ de duas maneiras:

a) Por meio da fórmula de Bhaskara e por meio do vértice da parábola.

Entrada

A entrada consiste de uma única linha que contém um inteiro, ***P*** ($0 \leq P \leq 500$) indicando a quantidade de passageiros no voo, dois valores de ponto flutuante, ***V*** ($0 \leq V \leq 10000$) indicando o valor da passagem por passageiro e ***D*** ($0 \leq D \leq 500$) indicando o desconto por passageiro adicional e um inteiro ***C*** ($0 \leq C \leq 500$) indicando a capacidade da aeronave.

Saída

Seu programa deve produzir duas linhas, uma apresentando o número de passageiros em cada avião, a fim de que se obtenha a maior receita possível? E outra com o valor da receita máxima?

Exemplos de Entradas	Exemplos de Saídas
`100 500.00 10.00 150`	`Passageiros: 75` `Receita maxima: 56250.00`
`40 350.00 5.00 60`	`Passageiros: 55` `Receita maxima: 15125.00`

Problema D
Fábrica de Automóveis

Por Murilo Varges da Silva
Arquivo: fabrica.[c|cpp|java]
***Timelimit:** 1*

Uma indústria fabrica modelos diferentes de automóveis, a primeira tabela mostra a quantidade de parafusos e travas utilizadas em cada modelo. A segunda tabela mostra a produção esperada para os meses de setembro e outubro.

	Modelos		
Peça	**A**	**B**	**C**
Parafusos	4	5	7
Travas	1	2	2

	Set.	Out.
A	300	400
B	500	600
C	200	300

Quantos parafusos e quantas travas serão necessários para a produção nesses dois meses?

	Set	Out
Parafusos	5100	6700
Travas	1700	2200

Você foi contratado para desenvolver um programa que calcula a quantidade de parafusos e travas necessários para a produção de cada mês dos modelos de carros da indústria.

Entrada

A primeira linha da entrada contém dois números inteiros ***M*** ($0 \leq M \leq 10$) indicando a quantidade de modelos que serão fabricados e ***Q*** ($0 \leq Q \leq 12$) indicando a quantidade de meses que estes modelos serão fabricados. As duas próximas linhas irão conter ***M*** inteiros cada uma, que vão indicar quantidade de parafusos e travas necessárias para cada modelo. As próximas ***M*** linhas vão conter ***Q*** inteiros indicando a quantidade fabricada de cada modelo por mês.

Saída

Seu programa deve produzir como saída duas linhas com ***Q*** colunas indicando a quantidade de peças necessárias a cada mês para produzir os modelos de carros. Como separador de colunas utilize o caractere <TAB> "\t" e no final de cada linha utilize o caractere final de linha "\n".

Exemplos de Entradas	Exemplos de Saídas
3 2 4 5 7 1 2 2 300 400 500 600 200 300	5100 6700 1700 2200
4 3 10 11 12 13 14 15 16 17 300 400 500 400 500 600 500 600 700 600 700 800	21200 25800 30400 28400 34600 40800

Problema E
Imposto de Renda

Por Luís Hideo Vasconcelos Nakamura
Arquivo: imposto.[c|cpp|java]
***Timelimit:** 1*

Um contador está cursando aulas de programação e decidiu pedir a sua ajuda com o trabalho final de curso. Ele quer criar um programinha fictício e simples, mas que ajude os leigos na hora de declarar um Imposto de Renda que ele considera justo. Esse programa deve identificar se o usuário deve ou não declarar o imposto de renda, ou seja, se o usuário é isento ou não isento. Em caso do usuário não ser isento, ele deve calcular o imposto a ser pago, conforme a tabela de salários abaixo (lembre-se que o imposto é recolhido anualmente; 12 salários + 13º salário). Porém, o usuário pode passar informações para deduzir o imposto, como: o número de dependentes e um valor gasto com saúde.

A cada dependente o programa de conceder um desconto de 10% sobre o imposto que deve ser pago, até o limite máximo de 30%. O valor de gasto com saúde também pode ser abatido do imposto de renda, sendo que esse valor pode ser descontado até o valor limite de 15% do total de imposto a ser pago. Ou seja, se o imposto a ser pago for de 1000 reais, e foram gastos com saúde 200 reais, o limite de desconto será 150 reais 15%. Porém, se ele gastou 100 reais com saúde a dedução será de apenas 100 reais. O contribuinte pode ter ambos os descontos, desde que os seus limites sejam respeitados.

BASE DE CÁLCULO (Salário Mensal)	ALÍQUOTA (Taxa de imposto %)
Até 2999 reais	Isento
De 3000 reais até 4999 reais	5%
De 5000 reais até 9999 reais	10%
De 10000 reais até 14999 reais	15%
De 15000 reais até 19999 reais	20%
De 20000 reais até 24999 reais	25%
De 25000 reais até 29999 reais	30%
Acima de 30000	35%

A entrada do usuário para o programa deve ser da seguinte forma:
>> [Salário mensal] [Nº de Dependentes] [Valor gasto com Saúde]
Exemplo, um usuário com salário mensal de 2000 reais, com dois dependentes e gastou 800 reais com saúde:
>> 2000 2 800
A saída do programa pode ser:
<< Isento
Ou
<< VALOR
Onde VALOR é valor do imposto a ser pago.
OBS Importante: considere que todos os valores de entrada e de saída não devem ter casas decimais ("%.0f").

Exemplos de Entrada	Exemplos de Saída
1000 3 900	Isento
2999 2 6000	Isento
3000 0 0	1950
3000 1 0	1755
3000 0 263	1687
5000 2 0	5200
5000 3 15000	3575

Problema F
Pipoqueiro

Adaptado Por João Paulo Lemos Escola
Arquivo: pipoqueiro.[c|cpp|java]
***Timelimit:** 1*

A fim de atrair torcedores para os jogos da 4ª divisão do campeonato municipal de futebol, a equipe da casa distribuiu vale-pipocas aos torcedores na entrada do estádio.

Isso se mostrou uma ótima oportunidade de lucro para o pipoqueiro Ceará. Por ser um pipoqueiro ambicioso, deseja saber qual a melhor posição que deve ficar durante os jogos a fim de lucrar o máximo possível.

Em um estádio com 3 arquibancadas, sabe-se que todo cliente demora 1 minuto para ir de uma arquibancada a outra.

Posicionando-se estrategicamente entre as 3 arquibancadas, Ceará poderá lucrar mais, por isso, ele quer que você o ajude, desenvolvendo um programa que calcule o tempo mínimo para o trajeto de seus possíveis clientes de acordo com a posição dele entre as arquibancadas.

Entrada

A entrada tem diversos casos de teste e consiste em 3 números E1, E2 e E3, de 0 a 1000, onde cada número representa a quantidade de pessoas com vale-pipocas em cada arquibancada.

Saída

A saída de seu programa deve ser o número total de minutos gastos no melhor posicionamento do pipoqueiro entre as arquibancadas.

Exemplos de Entrada	Exemplos de Saída
2 3 4	12
3 4 2	10

Problema G
BK em Birigui

Por Murilo Varges da Silva
Arquivo: bk.[c|cpp|java]
***Timelimit:** 1*

É apenas uma moda passageira ou está aqui para ficar? O número de lanchonetes "*fast-food*" que estão abrindo franquias em Birigui está crescendo de forma acelerada, este crescimento já despertou o interesse de duas das dez maiores redes de "*fast-food*" do mundo (*Subway e McDonald's*), que já iniciaram operação na cidade.

Diante desta tendência outros gigantes do seguimento estão de olho na cidade de Birigui, também conhecida como "A *Massachussets* Brasileira" por sediar os grandiosos jogos do lendário "Rock Gol" campeonato de futebol disputado por músicos, exibido pela MTV entre 1995 e 2008 que era apresentado por Paulo Bonfá.

Nesta briga de gigantes surge o BK - Burger King ou "*Rei dos Hambúrgueres*" que está de olho nesta oportunidade, eles estão interessados em se sediar em Birigui e precisam identificar locais onde já se concentram restaurantes deste tipo para tentar "roubar" a clientela dos concorrentes. Sua tarefa é a partir de um mapa da cidade, marcado com as localizações dos restaurantes "*fast-food*, encontrar o local mais próximo do maior número de restaurantes na cidade. Como você provavelmente já sabe, sua cidade é construída em um layout de quadras, com blocos alinhados em eixos norte-sul e leste-oeste. Deste modo você tem que caminhar ao longo das ruas, e a distância entre interseções (a, b) e (c, d) é | a - c | + | b - d |.

Entrada

A entrada contém vários casos de teste. Cada caso de teste descreve uma cidade. A primeira linha de cada caso de teste contém quatro inteiros ***dx, dy, n,*** e ***q***. Estas são as dimensões da cidade ***dx*** × ***dy*** ($1 \leq dx, dy \leq 1000$), o número de restaurantes "*fast-food*" ***n*** ($0 \leq n \leq 5 * 10^5$), e o número de consultas ***q*** ($1 \leq q \leq 20$). Cada uma das próximas ***n*** linhas contém dois inteiros x_i e y_i ($1 \leq x_i \leq dx$, $1 \leq y_i \leq dy$); eles especificam a localização do restaurante ¡enésimo. Haverá no máximo, um restaurante por cruzamento. Cada uma das próximas ***q*** linhas contém um único inteiro ***m*** ($0 \leq m \leq 106$), indicando a distância máxima entre um restaurante e um ponto de interesse. O último caso de teste é seguido por uma linha contendo quatro zeros.

Saída

Para cada caso de teste da entrada, exibir seu número do processo. Em seguida, exibir uma linha por consulta do caso de teste. Cada linha exibe o número máximo de restaurantes alcançáveis para a consulta de determinada distância ***m*** seguido do local ideal. Por exemplo, a saída de exemplo do "Caso 1" mostra que 3 restaurantes estão dentro da distância de consulta 1 e a localização ideal é (3,4), 4 restaurantes estão dentro da distância de consulta 2 e a localização ideal é (2,2) e 5 restaurantes estão a uma distância de consulta 4 e a localização ótima é (3,1). Se houver vários locais ideais, escolher o local que está mais ao sul (o menor inteiro positivo coordenada y). Se ainda houver um empate, escolher o local mais ocidental (o menor inteiro positivo coordenada x). Siga o formato da saída do exemplo.

Exemplos:

Entradas	Saídas
4 4 5 3 1 1 1 2 3 3 4 4 2 4 1 2 4 1 1 0 3 0 1 2 1 1 1 3 1 1 0 1 2 10 10 1 4 7 7 0 1 5 10 4 4 2 5 2 3 3 2 4 3 2 1 0 0 0 0 0	Caso 1: 3 (3,4) 4 (2,2) 5 (3,1) Caso 2: 0 (1,1) 0 (1,1) 0 (1,1) Caso 3: 1 (1,1) 1 (1,1) 1 (1,1) Caso 4: 1 (7,7) 1 (7,6) 1 (7,2) 1 (3,1) Caso 5: 2 (1,1) 2 (1,1) 2 (2,1) 2 (2,2) 1 (3,2)

Problema H
Teclado

Adaptado Por Murilo Varges da Silva
Arquivo: teclado.[c|cpp|java]
***Timelimit:** 1*

Quantas teclas são necessárias pressionar para escrever uma mensagem de texto? Você pode pensar que é igual ao número de caracteres no texto, mas isso só é correto se uma tecla gera um caractere. Com dispositivos de bolso, as possibilidades para a digitação de texto são muitas vezes limitadas. Alguns dispositivos fornecem apenas alguns botões, um número significativamente menor do que o número de letras no alfabeto. Para esses dispositivos, o usuário precisa apertar vários botões para digitar um único caractere. Um mecanismo para lidar com essas limitações é um teclado virtual exibido em uma tela, com um cursor que pode ser movido para selecionar caracteres. Quatro botões de seta controlam o movimento do cursor, e quando o cursor é posicionado sobre uma tecla apropriada, pressionando o quinto botão seleciona o carácter correspondente e anexa-o ao fim do texto. Para finalizar o texto, o usuário deve navegar e selecionar a tecla "Enter". Isso fornece aos usuários um conjunto arbitrário de caracteres e que lhes permite digitar texto de qualquer tamanho, com apenas cinco botões de hardware.

Neste problema, você terá um layout de teclado virtual e sua tarefa é determinar o número mínimo de teclas pressionadas para escrever um texto, onde pressionar qualquer um dos cinco botões de hardware conta como uma tecla pressionada. As teclas estão dispostas em uma tabela retangular, de tal modo que cada tecla virtual ocupa uma ou mais unidades da tabela. O cursor começa no canto superior esquerdo do teclado e move-se em quatro pontos cardeais, de tal forma que sempre avança para a próxima unidade quadrada em direção de uma tecla diferente. Se não existe tal quadrado de unidade, o cursor não se move.

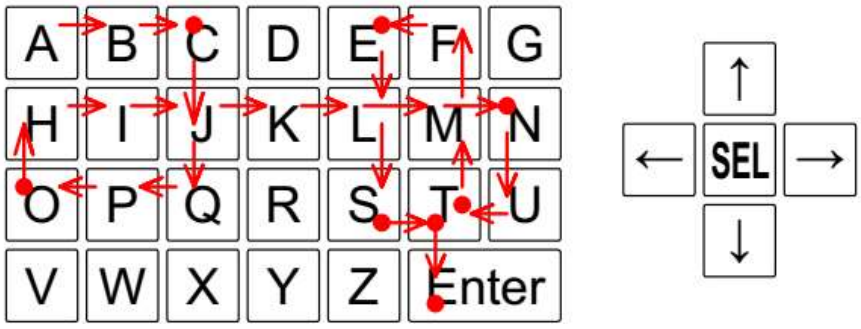

Figura 1 - Exemplo Entrada 1. Um exemplo de um teclado virtual e dos botões em hardware.

Figura 1, ilustra exemplo Entrada 1, mostra uma maneira possível de digitar a palavra "CONTEST" 30 toques em um teclado virtual como o da Figura 1. Os pontos vermelhos representam as teclas virtuais, onde o botão de seleção foi pressionado.

Entrada

A primeira linha da entrada contém dois números inteiros *r* e *c* ($1 \leq r, c \leq 50$), dando o número de linhas e colunas da grade do teclado virtual. O teclado virtual é especificado nas próximas *r* linhas, cada uma das quais contém *c* caracteres. Os possíveis valores desses caracteres são letras maiúsculas, dígitos, um traço e um asterisco (representando Enter). Há apenas uma tecla que corresponde a qualquer caractere dado. Cada tecla é constituída por um ou mais quadrados da tabela que representa o teclado, que irão sempre formar uma região ligada. A última linha da entrada contém o texto a ser digitado. Este texto é uma string não-vazia de, no máximo, 10.000 caracteres, sem asterisco.

Saída

Exibir o número mínimo de toques necessários para digitar o texto todo, incluindo a tecla “Enter” no final. Garantindo que o texto pode ser digitado.

Exemplos:

Entradas	**Saídas**
4 7 ABCDEFG HIJKLMN OPQRSTU VWXYZ** CONTEST	30

5 20 12233445566778899000 QQWWEERRTTYYUUIIOOPP -AASSDDFFGGHHJJKKLL* --ZZXXCCVVBBNNMM--** -------------------- ACM-ICPC-WORLD-FINALS-2015	160

2 19 ABCDEFGHIJKLMNOPQZY X*****************Y AZAZ	19

6 4 AXYB BBBB KLMB OPQB DEFB GHI* AB	7